IN THE CLAIMS

1. (Currently amended) A method comprising:

storing a plurality of partial keys corresponding to an equal number of original keys in ~~memory~~ a hash table, wherein storage of ~~said~~ the plurality of partial keys requires less memory than storage of ~~said~~ the equal number of original keys, and wherein ~~said~~ the plurality of partial keys are used to determine hashing conflicts;

applying a hash function to an original key of said ~~plurality~~ equal number of original keys to generate a partial key and a hash value, wherein the hash value includes a number of bits equal to a number of bits of the original key minus a number of bits of the partial key;

accessing the ~~memory~~ hash table according to the hash value;

reading a stored partial key of ~~said~~ the plurality of partial keys from the ~~memory~~ hash table that corresponds to ~~said~~ the hash value, wherein ~~said~~ the hash value is not stored in the ~~memory~~ hash table; and

executing a conflict check by comparing [[a]] the partial key ~~derived~~ generated from ~~an incoming full~~ the original key with the stored partial key ~~where the partial key corresponds at most with one of the stored partial keys~~.

2. (Currently amended) The method of Claim 1, wherein the partial key from the ~~memory~~ hash table corresponding to the hash value includes saved bits comprising a consecutive, sequential string of bits that is a subset of the original key where the subset includes a majority of bits of the original key.

3. (Previously presented) The method of Claim 2, wherein the stored partial key comprises a number of bits equal to or more than a number of bits of the original key minus a number of bits of the hash value.

4. (Original) The method of Claim 1, wherein the hash value is implemented by a linear feedback shift register.

5. (Previously presented) The method of Claim 1 further comprising applying a reverse function on the stored partial key and the hash value to generate the original key.

6. (Currently amended) The method of Claim 1 further comprising the steps of:

reading a result from the ~~memory~~ hash table corresponding to the hash value; and

forwarding a packet of data according to the result read from the ~~memory~~ hash table.

7. (Currently amended) An apparatus comprising:

a ~~memory~~ hash table which stores a plurality of partial keys used to determine hashing conflicts, wherein ~~said~~ the plurality of partial keys correspond to a plurality of original full keys~~, and wherein a data sum of said plurality of partial keys requires less memory than a data sum of said plurality of original keys~~;

a hash function block coupled to ~~a memory~~ the hash table that applies any polynomial to a full key and generates a partial key and a hash value which is used to point to one of the plurality of partial keys stores in the ~~memory~~ hash table, wherein the

plurality of partial keys include saved bits comprising consecutive, sequentially strings of bits derived from the plurality of original full keys, and wherein the hash value includes bits from the full key that are not included in any of the partial keys; and

a processor that compares one of the plurality of partial keys to the partial key comprising a majority of bits of the full key, wherein the hash value is not saved in the hash table.

8. (Currently amended) The apparatus of Claim 7, wherein the ~~memory~~ hash table comprises a hash table size.

9. (Currently amended) The apparatus of Claim 7, wherein the one of the plurality of partial keys stored in the ~~memory~~ hash table comprises a number of bits equal to or more than a number of bits of the full key minus a number of bits of the hash value.

10. (Previously presented) The apparatus of Claim 7, wherein the hash function block comprises a linear feedback shift register.

11. (Currently amended) The apparatus of Claim [[9]] 10, wherein the linear feedback shift register corresponds to a Galois version.

12. (Currently amended) The apparatus of Claim [[9]] 10, wherein the linear feedback shift register corresponds to a Fibonacci version.

13. (Currently amended) The apparatus of Claim 7 further including a reverse function generator coupled to the ~~memory~~ hash table, wherein the reverse function generator restores the full key based on the one of the plurality of partial keys stored in the ~~memory~~ hash table and the hash value.

14. (Currently amended) The apparatus of Claim 7 further comprising a forwarding engine coupled to the ~~memory~~ hash table, wherein the forwarding engine forwards a data packet according to information read from the ~~memory~~ hash table at an address corresponding to the one of the plurality of partial keys stored in the ~~memory~~ hash table.

15-26. Cancelled

27. (Currently amended) A method comprising:

generating a partial key and a hash value from an original key, where the partial key includes a consecutive subset of a majority of bits of the original key and the hash value includes a number of bits equal to a number of bits of the original key minus a number of bits of the partial key;

accessing a ~~memory~~ hash table including multiple partial keys, where the hash value is not stored in the ~~memory~~ hash table;

selecting a stored partial key from the ~~memory~~ hash table that corresponds with the hash value;

comparing the partial key with the stored partial key; and

identifying a hash conflict when the partial key matches the stored partial key.

28. (Currently amended) The method of claim 27 where the multiple partial keys correspond to an equal number of multiple input keys ~~and a data sum of all the multiple partial keys is less than a data sum of all the equal number of multiple input keys~~.

29. (Previously presented) The method of claim 28 where the multiple partial keys are each selectable according to a different hash value derived from one of the equal number of multiple input keys.

30. (Currently amended) The method of claim 27 where the comparing of the partial key ~~includes~~ comprises reading less data than that contained in the original key.

31. (Currently amended) The method of claim 32 where the hash value corresponds to a single entry in the ~~memory~~ hash table.

32. (Previously presented) The method of claim 27 including recovering the original key by combining the stored partial key with the hash value.

33. (Previously presented) The method of claim 32 where the original key is recovered by a reverse linear feedback shift register.

34. (Currently amended) A system comprising:

a hash function configured to generate a hash value and a partial key from an input key, where the partial key includes a consecutive sequential string of bits derived from the input key and the hash value includes a number of bits equal to a number of bits of the input key minus a number of bits of the partial key;

a ~~memory~~ hash table including stored partial keys that correspond to an equal number of input keys, where ~~a stored~~ the partial key includes less data bits than ~~its corresponding~~ the input key and where the hash value is not stored in the hash table; and

a processor configured to:

identify the ~~stored~~ partial ~~keys~~ key that is associated with the hash value;

compare the partial key to the stored partial ~~key~~ keys; and

identify a hash conflict when the partial key matches one of the stored partial ~~key~~ keys.

35. (Currently amended) The system of claim 34 where the stored partial ~~key includes~~ keys include a majority of data bits of ~~its~~ the corresponding input ~~key~~ keys.

36. (Currently amended) The system of claim 34 where the partial key includes multiple consecutive sequential strings of bits derived from the input key, separated by the bits of the hash value ~~is not stored in the memory~~.

37. (Currently amended) The system of claim 34 where the hash value includes an address location associated with the ~~corresponding~~ input key

38. (Currently amended) The system of claim 34 where the hash value includes bits from the input key that are not included in any of the stored partial keys ~~the memory includes a data sum of the stored partial keys which is less than a data sum of the equal number of input keys~~.